Impact of Parallelism on Dualcore

Varsha Thakur
Research scholar
Pt.Ravishankar shukla
university,Raipur(C.G),India

Sanjay Kumar
Associate Professor
Pt.Ravishankar shukla
university,Raipur(C.G),India

ABSTRACT
This paper shows the effect of parallelism in multicore architecture. Performance was evaluated on the basis of the execution time of matrix multiplication between sequential algorithm and parallel algorithm in multicore processors. To implement matrix multiplication algorithms C programming language with OpenMp Libraries was used under Linux environment.

General Terms
Parallel and distributed computing.

Keywords
Matrix Multiplication, Parallel Algorithm, OpenMp

1. INTRODUCTION
A Multicore processor is an Integrated Circuit in which more than one processor or core are included for performance improvement and Simultaneous processing of parallel jobs. Matrix multiplication is a well known mathematical term used in a linear algebra. Many other important matrix problems can be solved via matrix multiplication, e.g., finding the N th power, the inverse, the determinant and eigenvalues etc. We are living in the era of parallel computing where performance and efficiency are of fundamental importance [1]. OpenMp is used for parallelizing the sequential matrix multiplication. In rest of paper we have define some basic concept of OpenMP, Multicore Architecture and Matrix Multiplication Algorithms.

OPENMP
OpenMp is an API (Application Program Interface) that use multithreaded and shared memory parallelism. Openmp is basically divided into three parts Compiler directives, runtime library routines and environment variable. It is an open specification for multiprocessing. OpenMp worked as a fork-join model where fork is master thread that use to create a team of parallel thread and join is used when the team of parallel threads complete their task they synchronize and terminate and left the master thread to execute sequential program. OpenMp visualize as parallel programming model on multicore architecture [3].

Multicore Architecture

A multicore places multiple processors on a single chip and each processor is called a core [2]. As we increase the capacity of chip placing multiple processors on a single chip became practical. These architectural designs are known as Chip Multiprocessors (CMPs), chip Multiprocessors are known as Multicore. A multi-core processor is a single with two or more independent processors. The instructions on multicore are ordinary CPU instructions, but the multiple processors can run multiple processes parallel at the same time by increasing the overall speed of the programs. Multicore span threads which divide the tasks between cores. It can execute multiple tasks at single time. Multicore is shared memory processors, all processors shares the same memory. Multicores are becoming popular for both server and desktop processors. By the next decade, it is expected to have processors with hundreds of cores on a chip.

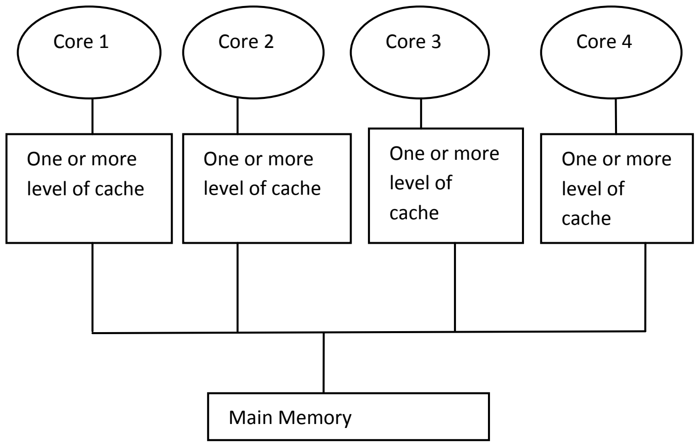


Fig 1. Multicore Architecture

2. MATRIX MULTIPLICATION
Matrix multiplication is a mathematical binary operation that takes input as pair of matrices, and gives output of another matrix.

2.1 Sequential matrix Multiplication
The sequential matrix multiplication is the fundamental basis for other matrix multiplication. Matrix multiplication is only possible when width of first matrix match with height of second matrix. The product of a X b matrix A with b X c matrix B is an a X c matrix C where element is defined as

$$C_{ij} = \sum^{b-1} a_{ik} b_{kj} \quad \text{where} \quad 0 \le i < a,\ 0 \le j < c$$

Sequential matrix multiplication requires a * b*c addition and same number of multiplication so, time complexity of multiplication of matrix using sequential algorithm is O (N^3).

2.2 Parallel Matrix Multiplication
In last few decades various approaches has been proposed for implementation of matrix multiplication on shared memory architecture. All parallel algorithms are based on conventional sequential matrix multiplication. For parallel matrix multiplication consider two nxn matrix A and matrix B. Partition the matrix in L blocks where (0<=i, j<root l) of size (n/root l) x (n/root l) each small matrix the n this small matrix mapped into root l X root l mesh of processors. The process initially stores Aij and Bij and compute Cij of result matrix. After computing the entire sub matrix, matrix A's block performed in each row and matrix B's performed in each coloum. Finally sub matrix multiplication and addition is performed. In parallel algorithm each element of matrix C is

computed simultaneously. Time complexity of multiplication of nxn matrix using parallel algorithm is O (N^2).

3. PERFORMANCE MEASUREMENTS

Performance of a parallel algorithm is measured using two factors speed-up and efficiency.

A. Speedup

In parallel computing, speedup refers to how much faster a parallel algorithm is run in parallel.

$$\text{Speed up} = \frac{\text{Sequential execution time}}{\text{Parallel execution time}}$$

Speed up depends on the ratio of the amount of time your code spends communicating to the amount of time it spends computing.

B. Efficiency

In parallel computing, efficiency refers to speed up divided by number of processors. Efficiency is a measure of how much of your available processing power is being used.

$$\text{Efficiency} = \frac{\text{Sequential execution time}}{\text{Parallel execution time X processor used}}$$

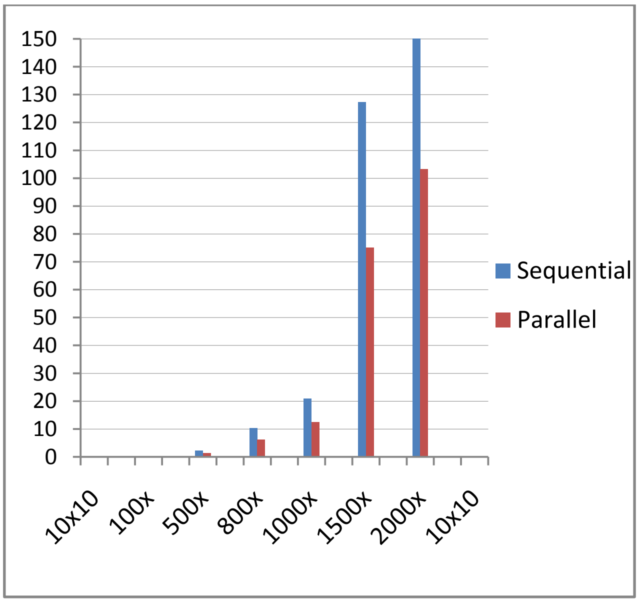
4. EXPERIMENTAL SETUP

For Experiment a computer systems are taken with dual core processor with 1.83 GHz speed and linux operating system. We have run the sequential matrix multiplication and parallel matrix multiplication and on dual core processor. Execution time was recorded as shown in Table I for dual core and analyzed graphically.

Table 1: Speed up of algorithms for dual core processors.

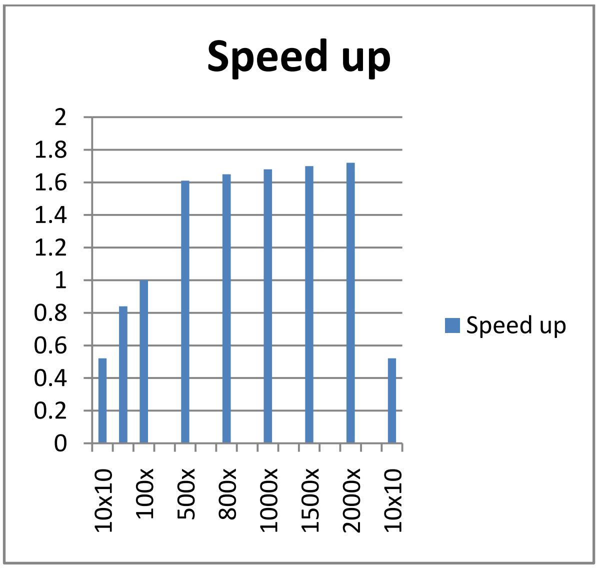
Matrix Size	Sequential(ms)	Parallel(ms)	Speed up
10x10	.0012	.0023	0.521
50x50	0.06	0.071	.84
100x 100	0.01	.01	1.0
500x 500	2.23	1.38	1.61
800x 800	10.34	6.24	1.65
1000x 1000	20.85	12.41	1.68
1500x 1500	127.3	75.14	1.70
2000x 2000	177.1	103.26	1.72
10x10	.0012	.0023	0.521

Graph 1 was plotted for time taken during execution of sequential and parallel algorithm in Linux platform in dual core processor.



The horizontal axis represents size of matrix and vertical axis represents execution time in milliseconds.

Graph 2 was plotted for Speedup in Linux platform in dual core processor.



The horizontal axis represents size of matrix and vertical axis represents execution time in milliseconds.

5. CONCLUSION

In this experiment execution time and speedup was calculated for sequential and, parallel matrix multiplication. It is clear from the graph as OpenMp parallaize sequential program performance get increases for higer order matrix while for lower order matrix execution time with parallalization is more than sequential multiplication. Beyond a certain optimum problem size only parallization is effective below that point because of communication overhead sequential algorithm on sequential machines will give better results. Below optimum problem size following overheads like interprocess communication overhead, synchronization and concurrency prominently play their roles.

6. REFERENCES

[1] Keqin L., Yi P., Si Qing Z.,"Fast and Processor Efficient Parallel Matrix Multiplication Algorithms on a Linear Array With a Reconfigurable Pipelined Bus System", IEEE Transaction on parallel and distributed, VOL. 9, AUG 1998, pp 705-720.

[2] Cameron, H., Tracy, H., Professional Multicore programming, Wiley publication,2008.

[3] Venkatesan P., Harish B., S. Sarholz, Proceedings of the 3rd international workshop on OpenMP "A Practical Programming Model for the Multi-core Era", 2008.

[4] Rose M. P., "A Parallel Approach for Matrix Multiplication on the TMS320C4x DSP", Digital Signal Processing Semiconductor Group, Texas Instruments**,** Feb 1994.

[5] http://en.wikipedia.org/wiki/Matrix_multiplication

[6] http://en.wikipedia.org/wiki/Strassen_algorithm

[7] Juby M., R. Vijaya K., "Comparative Study of Strassen's Matrix Multiplication algorithm", International Journal of Computer Science And Technology IJCST Vol. 3, Issue 1, Jan. - March 2012.

[8] Sara R.,"Toward an Optimal Algorithm for Matrix Multiplication", From SIAM News, Vol. 38, No. 9, November 2005.

[9] Quin Michael J."Parallel programming in C with MPI and OpenMP". McGraw Hill Inc., 2004.

[10] .Kai hwang, Naresh Motwani, Advanced computer Architecture, pp 108 chapter 3

[11] Jaeyoung C., A New Parallel Matrix Multiplication Algorithm on Distributed-Memory Concurrent Computers

[12] Vijayalakshmi S., Mohan R., A.S.Basavesh ,"A Comparative Study on Performance Benefits of Multi-core CPUs using OpenMP"IJCSI International Journal of Computer Science Issues, Vol. 9, Jan 2012.